



CHOUETTE

Maintenance, accompagnement et recette de logiciels pour les échanges de données multimodales

Manuel d'Installation

-

application Chouette V3.0

Auteurs :	Michel ETIENNE, Luc DONNET, Marc FLORISSON (CityWay)
Relecteurs	Patrick GENDRE (CEREMA), Jean SENG (AFIMB)

Résumé :

CHOUETTE est un logiciel libre développé à l'initiative du ministère français chargé des transports (et du développement durable) dans le but de faciliter l'échange de données d'offre (théorique) de transport collectif (TC), en s'appuyant pour cela sur la norme NFP 99506, dite Neptune, qui spécifie un profil d'échange XML.

Les utilisateurs visés sont les collectivités locales Autorités Organisatrices de Transport (AOT), les exploitants des réseaux TC, ainsi que leurs prestataires (bureaux d'étude ou société de services) et d'autres acteurs tels que services de l'état, éditeurs de logiciels, opérateurs de services d'information, chercheurs...

L'application CHOUETTE est disponible sous la forme d'une application WEB destinée à être déployée sur un serveur WEB, et d'une application en ligne de commande.

Ce manuel d'installation liste les logiciels à installer, puis les instructions pour créer la base de données, installer et configurer l'application web.

Agence française pour l'information multimodale et la billettique**Informations sur ce document :**



Organisme commanditaire : AFIMB			
Titre : Manuel d'installation de l'application CHOUETTE (code source V3.0)			
Sous-titre : Procédure à suivre pour installer et configurer l'application CHOUETTE.			
Organismes auteurs CITYWAY CEREMA DT Med.		Rédacteurs Marc FLORISSON Zakaria BOUZIANE Michel ETIENNE Luc DONNET	Participants Patrick GENDRE Jean SENG
Maitre d'ouvrage AFIMB			
Mots clés : profil d'échange Neptune, information multi-modale, application Chouette, manuel d'installation, PostgreSQL, JAVA , Ruby		Diffusion :  publique (licence Creative Commons CC-by-nd)	
Nombre de pages : 15 pages	Date Juillet 2015	Confidentialité : Non	Bibliographie : Oui
Version du document : 1.3		Versions du code source applicable : V3.0.x	

Historique des versions / révisions :

Version	Date d'application	Description des changements	Auteur
1.0	Mai 2015	Serveur IEV	Michel Etienne
1.1	Juin 2015	Compléments sur le Serveur IEV	Michel Etienne
1.2	Juin 2015	Compléments sur l'application Web	Luc Donnet
1.3	Juillet 2015	Précisions sur l'installation du serveur IEV	Michel Etienne



Table des matières

1Vue d'ensemble.....	4
2Prérequis.....	5
2.1.Logiciels.....	5
2.2.Dépendances externes.....	6
3Migration depuis une version précédente.....	7
3.1.Version V1.x vers une version 2.x.....	7
3.2.Version V2.x vers V3.0.....	7
4Installation.....	7
4.1.Récupération des sources.....	7
4.2.Installation des bases de données.....	8
4.3.Installation du serveur WEB.....	8
4.4.Installation du serveur IEV.....	9
5Configuration du serveur IHM WEB.....	11
6Lancement des serveurs.....	12
6.1.Serveur IHM WEB.....	12
6.2.Serveur IEV.....	13
7Annexes: configuration de l'installation Chouette IHM WEB.....	13
7.1.Paramétrage de l'accès aux cartes du géoportail IGN.....	13
7.2.Cas d'un déploiement sans serveur de messagerie e-mail.....	14



1 VUE D'ENSEMBLE

Ce document décrit comment installer l'application Chouette, mais également comment l'utiliser depuis un interpréteur Ruby pour manipuler les données métier.

Java et Ruby étant portables, Chouette est compatible avec de multiples plates-formes : Mac, Linux, Windows, etc. Le présent manuel s'adresse à un utilisateur confirmé ; il décrit en détail la procédure d'installation pour une plate forme Linux de type Debian.

Une documentation de base (« quick install ») des commandes linux à lancer pour installer Chouette est disponible en ligne : <https://github.com/afimb/chouette2/blob/V3.0/README.md>

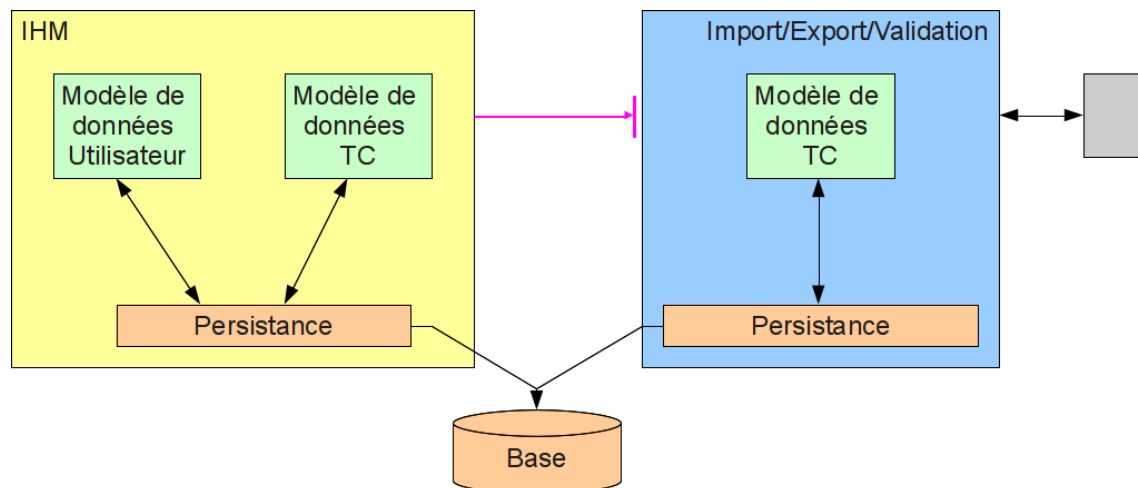
La version de linux doit être basée sur un noyau version 2.6 minimum et 64 Bits. (C'est le cas par exemple Ubuntu 13.10).

Sous Debian 7.5 et + (et ubuntu 13 et+), Chouette peut aussi être installé comme Package :

cf. <http://packages.chouette.mobi/debian/chouette/>

Au niveau mémoire, il est recommandé de disposer d'une machine ayant au minimum 2 Go de RAM. Pour traiter de grands réseaux comme ceux d'Ile de France, il est nécessaire de disposer d'un minimum de 4Go de Ram et de 4 CPUs (mais attention dans ce cas à utiliser une JVM 64 bits).

Chouette est une application web organisée selon l'architecture suivante :



L'application Chouette comprend :

- un serveur web Ruby/Rails pour gérer l'IHM (fonctions de gestion des données) et pour piloter les actions d'import, d'export et de validation
- Un serveur Rest (serveur IEV) en Java pour exécuter les actions d'import d'export et de validation

Les deux serveurs se partagent une base de données PostgreSQL commune pour lire et sauvegarder les données de transport



L'installation de l'application nécessite 4 étapes, détaillées dans les Chapitres suivants :

- 1- d'avoir installé les librairies et outils (paquets) utilisés par le logiciel
- 2- d'installer la base de données
- 3- de générer ou récupérer les serveurs compilés (ear)
- 4- d'installer et de configurer ces serveurs

2 PRÉREQUIS

2.1. Logiciels

Le tableau ci-dessous dresse la liste des logiciels nécessaires avant de procéder à l'installation de Chouette. Ces composants doivent être installés dans votre environnement suivant leurs propres procédures (si nécessaire, reportez-vous aux procédures et/ou outils d'installation de ces composants)

Logiciel	Version minimale	Version conseillée	Intérêt
Plate forme JRE http://www.java.com/fr/download	1.7 (64 bits)	1.7 (64 bits)	Exécution en ligne de commande
Wildfly http://wildfly.org/	8.2.0-Final	8.2.0-Final	Serveur d'application J2E
Ruby https://www.ruby-lang.org/fr/	2.1.6	2.1.6	Interpréteur Ruby pour l'application WEB
Base PostgreSQL http://www.postgresql.org/	9.3	9.3	Persistence des données de l'application CHOUETTE
Driver jdbc postgresql https://jdbc.postgresql.org/download.html	9.3-1103	9.3-1103	Pilote d'accès à Postgresql à installer dans Wildfly
Librairie libpq	9.3	9.3	API postgresql
Bundler (gem ruby) http://bundler.io/	1.10.3	1.10.3	Gestionnaire de l'environnement d'une application Ruby
Proj.4 http://trac.osgeo.org/proj/	4.8.0	4.8.0	Fonctions Géographiques : - pour Linux : packets proj-bin et libproj-dev - pour Windows : proj_lib



Librairie Proj4

Note : Sous windows, une fois l'installation de OSGeo4W64 effectuée :

- créer la variable d'environnement PROJ_LIB = [répertoire OSGeo4W]\share\proj
- ajouter au Path : [répertoire OSGeo4W]\bin

2.2. Dépendances externes

En complément des logiciels nécessaires au déploiement, si vous souhaitez générer les exécutables à partir des sources, il faudra en outre installer les logiciels suivants pour linux (sous Windows nous n'avons pas documenté la génération des exécutables).

Logiciel	Version minimale	Version conseillée	Intérêt
Plate forme jdk http://www.oracle.com/technetwork/java/javase/downloads/index.html	1.7	1.7	Compilation java
Git pour récupérer les modules sur github http://git-scm.com/	1.7.9	1.7.10	Accès au référentiel des sources de Chouette
Maven	3	3	Compilation java
Unzip			Décompression des archives
make			Installation des gems

Exemple d'installation des paquets sous Linux base Debian :

```
sudo apt-get install postgresql-9.3
sudo apt-get install libpq-dev
sudo apt-get install openjdk-7-jdk
sudo apt-get install git
sudo apt-get install unzip
sudo apt-get install proj-bin
sudo apt-get install libproj-dev
sudo apt-get install make
sudo apt-get install maven
```



Installer RVM soit par paquet si la distribution linux en dispose

```
sudo apt-get install ruby-rvm
```

Sinon Installer RVM en suivant la procédure :

```
sudo apt-get install curl
gpg --keyserver hkp://keys.gnupg.net --recv-keys
409B6B1796C275462A1703113804BB82D39DC0E3
\curl -sSL https://get.rvm.io | bash -s stable
```

Installer ruby 2.1.6 à partir de RVM

```
rvm install ruby-2.1.6
rvm --default use 2.1.6
```

Installer le gem « bundler » ou le mettre à jour

```
gem install bundler -v 1.10.3
```

Installer wildfly : (dans un répertoire du style /opt/wildfly), exemple d'installation simplifiée :

```
wget http://download.jboss.org/wildfly/8.2.0.Final/wildfly-8.2.0.Final.zip
unzip wildfly-8.2.0.Final.zip
```

3 MIGRATION DEPUIS UNE VERSION PRÉCÉDENTE

3.1. Version V1.x vers une version 2.x

La version 2 correspond à une réécriture complète de l'IHM dans le framework Ruby on Rails. Le schéma de la base 2.0 est totalement modifié pour s'adapter aux règles de Rails (Active Records); de plus, la gestion des organisations, des utilisateurs, et des espaces de données ne permettent pas de réaliser une procédure de migration de schéma..

Pour récupérer vos données gérées avec une version antérieure à la V2 dans la nouvelle version, il faut donc exporter l'ensemble des réseaux au format Neptune, puis importer les fichiers Neptune dans les espaces de données adéquats depuis Chouette V2.

3.2. Version V2.x vers V3.0

La procédure d'installation décrite ci-après convient aussi bien pour une première installation que pour une mise à jour de version.

La descente de version n'est pas garantie (il peut y avoir des pertes de données)

Note : Le passage à la version 3.0 détruit les rapports d'import, d'export et de validation existants.

4 INSTALLATION

L'installation décrite ci-après suppose d'être faite avec un compte utilisateur linux dédié (exclure root) , celui-ci est nommé `myuser:mygroup` dans la suite du document.



Ce compte doit disposer du droit 'sudoer'

4.1. Récupération des sources

Cette section présente la procédure pour générer les exécutables du serveur web et du serveur iev, utilisé par le premier pour les imports / exports / validations. Il est rédigé **uniquement pour linux DEBIAN et systèmes dérivés** ; la procédure devrait fonctionner pour Windows compte tenu de la portabilité de Java et Ruby, mais elle n'est pas documentée ici.

Avant de télécharger les sources, nous vous invitons à nous laisser vos coordonnées sur le site internet <http://www.chouette.mobi>, rubrique Club Utilisateurs, ce qui nous permet de mieux connaître nos utilisateurs.

L'archive contenant les sources est accessible aux adresses :

<http://github.com/afimb/chouette> : code java du serveur IEV (branche V3_0)

<http://github.com/afimb/chouette2> : code ruby du serveur IHM web (branche V3_0)

Attention : les versions stables de Chouette et Chouette2 sont celles qui portent un TAG ; se référer au dernier TAG disponible sur Github. La version à date d'édition de ce document est la V3.0.0 pour Chouette et Chouette2.

Note : chouette peut être récupérée sous forme binaire à l'adresse suivante :

http://maven.chouette.mobi/mobi/chouette/chouette_iev/x.y.z/chouette_iev-x.y.z.zip

(où x.y.z représente la version de l'application)

4.2. Installation des bases de données

Les serveurs Web et IEV utilisent une base Postgres, il est nécessaire de la configurer avant d'installer les serveurs ; pour une mise à jour cette étape n'est pas nécessaire

Pour une première installation : créer le compte utilisateur de la base de données

```
sudo su - postgres
createuser -s -d -P -U postgres chouette
Enter password for new role: chouette
Enter it again: chouette
exit
```

NOTE : la version 3 de Chouette oblige à passer Postgres en version 9.3 minimum, une migration de Postgres vers la 9.3 est donc à effectuer si nécessaire.

4.3. Installation du serveur WEB

1- Récupération des sources sous github

```
cd [working_directory]
wget https://github.com/afimb/chouette2/archive/V3.0.0.zip
unzip V3.0.0.zip
```



2- Télécharger les librairies externes (gems) :

```
cd chouette2  
  
bundle install
```

3- Initialiser ou mettre à jour la base de données

```
RAILS_ENV=production bundle exec rake db:create db:migrate
```

4- Préparer les données statiques

```
RAILS_ENV=production bundle exec rake assets:clobber assets:precompile
```

4.4. Installation du serveur IEV

Le serveur chouette_iev nécessite l'installation et la configuration du serveur d'applications J2E Wildfly et la récupération du driver jdbc postgresql

Créer et installer les bases IEV et Chouette :

```
sudo su - postgres  
createdb -O chouette iev  
createdb -O chouette -E UTF-8 -T template1 chouette2
```

Si le serveur est installé à partir des sources

```
sudo su - postgres  
  
createdb -O chouette -E UTF-8 -T template1 chouette_test
```

Adapter la mémoire du serveur Wildfly

Si le serveur est amené à traiter des lignes de transport volumineuses , passer la JVM de Wildfly de 512 Mo de mémoire à 1024Mo :

- éditer le fichier [Wildfly_install_directory]/bin/standalone.conf et modifier l'option -Xmx512m de JAVA_OPT

Lancement du serveur Wildfly (dans un terminal dédié)

```
cd [Wildfly_install_directory]  
bin/standalone.sh -b 0.0.0.0 -c standalone-full.xml [-Djboss.socket.binding.port-offset=100]
```

l'option jboss.socket.binding.port-offset permet de décaler les ports utilisés par Wildfly ; par défaut Wildfly utilise les port 8080 pour les applications et 9990 pour l'administration

Créer un compte administrateur (sur le terminal principal)

```
bin/add-user.sh
```



```
type: management user (a)
login : admin
password : admin
```

Note : Les comptes et mots de passe ne sont pas imposés.

Créer les connecteurs bases de données :

```
wget https://jdbc.postgresql.org/download/postgresql-9.3-1101.jdbc41.jar
bin/jboss-cli.sh

connect

module add --name=org.postgres --resources=(path to driver)/postgresql-9.3-1101.jdbc41.jar --dependencies=javax.api,javax.transaction.api

/subsystem=datasources/jdbc-driver=postgres:add(driver-name="postgres",driver-module-name="org.postgres",driver-class-name=org.postgresql.Driver)

data-source add --jndi-name=java:jboss/datasources/chouette --name=chouette --connection-url=jdbc:postgresql://localhost:5432/chouette2 --driver-name=postgres --user-name=chouette -password=chouette --max-pool-size=30

data-source add --jndi-name=java:jboss/datasources/iev --name=iev --connection-url=jdbc:postgresql://localhost:5432/iev --driver-name=postgres --user-name=chouette -password=chouette

/subsystem=ee/managed-executor-service=default/ :write-attribute(name=max-threads,value=<thread_size>)

/subsystem=ee/managed-executor-service=default/ :write-attribute(name=queue-length,value=<queue_size>)

reload
```

(path to driver) est à remplacer par le chemin absolu du driver postgresql téléchargé

<thread_size> et <queue_size> sont à adapter aux paramètres iev.started.jobs.max et iev.copy.by.import.max définis dans le fichier iev.properties :

- <thread_size> ≥ 2 x iev.started.jobs.max
- <thread_size> + <queue-size> ≥ iev.started.jobs.max x (iev.copy.by.import.max + 1)
- *Note : Les comptes et mots de passe ne sont pas imposés mais doivent rester cohérents avec l'installation du serveur web*

Paramétrage de la taille maximale des fichiers téléchargés

Par défaut Wildfly limite la taille des fichiers d'import à 10 Mo

pour l'ajuster aux besoins (exemple avec 80Mo pour être en adéquation avec les besoins du serveur Web) :

```
bin/jboss-cli.sh
```



```
connect

/subsystem=undertow/server=default-server/http-listener=default/ :write-at-
tribute(name=max-post-size, value=83886080)

exit
```

Paramétrage du répertoire de stockage et du nombre maximum de jobs

Le répertoire de stockage par défaut est USER_HOME ; pour le changer, il faut installer un fichier iev.properties dans le répertoire /etc/chouette/iev ; le contenu du fichier est le suivant (exemple fourni dans le zip de chouette_iev)

```
# properties for iev server
# base directory for referential storage (comment to use default location)
iev.directory=/var/lib/chouette/iev
# maximun jobs working at same time
iev.started.jobs.max=5
# maximun copy by import (save passing times for a line)
iev.copy.by.import.max=5
# admin key (comment or empty disable admin API)
iev.admin.key=
```

Installer le fichier iev.properties :

```
sudo mkdir -p /etc/chouette/iev
sudo chown -R <myuser>:<mygroup> /etc/chouette
cat > /etc/chouette/iev/iev.properties
# properties for iev server
# base directory for referential storage (comment to use default location)
iev.directory=/var/lib/chouette/iev
# maximun jobs working at same time
iev.started.jobs.max=5
# maximun copy by import (save passing times for a line)
iev.copy.by.import.max=5
# admin key (comment or empty disable admin API)
iev.admin.key=
Ctrl-C
```

Créer le répertoire de stockage :

```
sudo mkdir -p /var/lib/chouette/iev
```



```
sudo chown myuser:mygroup /var/lib/chouette/iev
sudo chmod a+x /var/lib/chouette/iev
```

Activation du l'API de supervision

L'api de supervision n'est pas active par défaut, il faut préciser dans le fichier `/etc/chouette/iev/iev.properties` le paramètre `iev.admin.key` et utiliser la même valeur dans la requête de supervision (paramètre `key` dans l'url).

```
# properties for iev server
# base directory for referential storage (comment to use default location)
iev.directory=/var/lib/chouette/iev
# admin key (comment or empty disable admin API)
iev.admin.key=MyKey
```

Installation du serveur IEV à partir des sources

```
cd [working_directory]
wget https://github.com/afimb/chouette/archive/V3.0.0.zip
unzip V3.0.0.zip
cd chouette
```

dans le répertoire racine créé par git : (wildfly doit être actif)

```
mvn clean install
```

la dernière étape publie l'application dans wildfly

Installation du serveur IEV à partir de l'archive compilée

```
wget
http://maven.chouette.mobi/mobi/chouette/chouette\_iev/3.0.0/chouette\_iev-3.0.0.zip
unzip chouette\_iev-3.0.0.zip
bin/jboss-cli.sh connect, deploy --force (path to ear)/chouette.ear
```

(path to ear) est à remplacer par le chemin absolu de l'archive EAR du serveur

5 CONFIGURATION DU SERVEUR IHM WEB

Les logs de l'application configurée par défaut sont redirigées dans les logs systèmes : `/var/log/syslog` ou `/var/log/messages` selon les distributions Linux.

La configuration **chouette2** est définie dans les fichiers :

- `config/environments/production.rb`
- `config/secrets.yml`



- config/initializers/devise_async.rb

Pour les différents paramétrages, il faut se référer aux documents suivants :

- Action Mailer : http://guides.rubyonrails.org/action_mailer_basics.html
- API Géoportail : <http://api.ign.fr/accueil>
- API Google Analytics : <https://www.google.fr/intl/fr/analytics/>
- Rails documentation : http://guides.rubyonrails.org/4_1_release_notes.html#config-secrets-yml
- Devise Async specification : <https://github.com/mhfs/devise-async>

D'autres détails sur la configuration des clés IGN et du mailer en l'absence de serveur de messagerie sur la machine sont données en Annexe.

Dans le fichier production.rb :

- Configuration des URLs des vues Action Mailer : Modifier le paramètre config.action_mailer.default_url_options
- Configuration de SMTP : Modifier le paramètre ActionMailer::Base.smtp_settings
- Configuration de l'adresse E-mail serveur : Modifier le paramètre config.mailer_sender

Dans le fichier secrets.yml (choisir au préalable le bon environnement)

- Configuration de la Clé Géoportail IGN : Décommenter et fixer le paramètre geoportal_api_key
- Configuration de Google Analytics : Modifier le paramètre google_analytic_tracker
- Configuration de l'url d'accès à l'API Chouette Web Service IEV : Modifier le paramètre api_endpoint
- Configuration de la clé pour Rails et Devise : Modifier le paramètre secret_key_base

Dans le fichier config/initializer/devise_async.rb

- Configuration de la façon dont l'application envoie les mails (non asynchrone par défaut mais peut être activé) : Modifier le paramètre Devise::Async.enabled

6 LANCEMENT DES SERVEURS

Les serveurs pourront être lancés au démarrage de la machine en installant des scripts dans les répertoires adaptés au système d'exploitation (Base RedHat ou Debian).

Dans la suite, le lancement est expliqué pour un démarrage à la demande.

6.1. Serveur IHM WEB

1- Lancer la gestion des tâches asynchrones pour l'envoi de mail si besoin (Voir [#5.Configuration du serveur IHM WEB|outline](#)) :



```
RAILS_ENV=production bundle exec rake jobs:work
```

Attention, cette commande ne rend pas la main

2- Lancer le serveur Web : (WEBrick : serveur par défaut de RubyOnRails)

```
RAILS_ENV=production bundle exec rails server
```

Cette action peut être remplacée par un déploiement sous Phusion Passenger sous Apache ou sous Nginx.

Apache et Nginx gèrent les ressources statiques, auquel cas il faut désactiver cette action dans `config/environments/production.rb` en passant le paramètre `serve_static_assets` à 'true'

Un document séparé présente l'utilisation de la console Rails pour accéder aux données via les objets métier.

6.2. Serveur IEV

1- Lancer le conteneur d'application Wildfly

```
cd [Wildfly_install_directory]
bin/standalone.sh -b 0.0.0.0 -c standalone-full.xml [-Djboss.socket.binding.port-offset=100]
```

l'option `jboss.socket.binding.port-offset` permet de décaler les ports utilisés par Wildfly ; par défaut Wildfly utilise les port 8080 pour les applications et 9990 pour l'administration

2- Créer des référentiels pour un usage indépendant du serveur web :

recupérer le fichier `chouette.sql` sous github :

https://github.com/afimb/chouette/blob/V3_0/src/main/sql/chouette.sql

créer un référentiel :

```
psql -h 127.0.0.1 -U chouette -v SCH=<schema_name> -d chouette2 -f
chouette.sql
```

où `<schema_name>` est le nom du référentiel (uniquement en minuscules et sans espace)

7 ANNEXES: CONFIGURATION DE L'INSTALLATION CHOUETTE IHM WEB

7.1. Paramétrage de l'accès aux cartes du géoportail IGN

L'accès aux cartes du géoportail nécessite une clé. C'est l'IGN définit la politique de distribution des clés d'accès et de leurs conditions d'utilisation :



- le site <http://api.ign.fr/accueil> permet d'obtenir une clé de développement (pour un serveur local accessible en <http://localhost>)
- le site <http://professionnels.ign.fr/> permet d'obtenir une clé de production (pour un site WEB publié sur un nom de domaine)

Pour un déploiement de l'application en environnement de production (au sens Rails), le fichier de paramétrage (`config/secrets.yml`) permet de définir la clé d'accès au portail IGN dans la rubrique production :

```
geoportail_api_key = « abcde »
```

7.2. Cas d'un déploiement sans serveur de messagerie e-mail

Dans `production.rb` :

retirer la structure `ActionMailer::Base.smtp_settings` ;

ajouter la variable `config.action_mailer.delivery_method = :letter_opener`